SMART INTERNZ PROJECT

ON

CAR PERFORMANCE PREDICTION USING IBM WATSON MACHINE LEARNING

Submitted by

RAHUL KOLAY (19BEC0169)
BHARATH P NAIR(19BEC0180)
YASHWANTH KOLLI (19BEC0149)
AMRISHA BHARDWAJ

1. INTRODUCTION

1.1 Overview:

Predicting the performance level of automobiles is a challenging and fascinating topic. The major purpose is to predict the car's performance in order to optimize the vehicle's specific behavior. This can considerably reduce fuel usage and increase efficiency in the system.

The car's performance is evaluated based on the engine type, number of cylinders, fuel type, and horsepower, among other factors. These are the variables that can be used to forecast the car's health. It is a continuous process of gathering, studying, analyzing, and documenting information about one's health based on the three variables listed above. Mileage, dependability, flexibility, and affordability are all performance objectives that can be paired together to help the prediction engine and engine management system. This method is critical for fully comprehending the vehicle's performance.

1.2 Purpose:

Prior to the last decade, cars were constructed for high-speed operation, comfort, and safety due to a surplus of gasoline. As the scarcity of fuel grows due to the excessive use of fuel in automobiles, numerous researchers have begun to study alternative fuels, car body redesign, and aerodynamic loss reduction. To compensate for these losses, spoilers are employed, necessitating the optimization of its shape. This research is primarily

focused on determining the optimal design of a car spoiler in order to reduce mass and hence fuel consumption while maintaining aerodynamic qualities and strength. The results of a Computational Fluid Dynamics (CFD) analysis of a two-dimensional model of a spoiler are validated by previous research in this field for understanding changes in aerodynamic property of cross-section. The results of the three-dimensional CFD analysis of the spoiler provide aerodynamic properties and pressure data that can be used to compare the results of the optimized model generated by the optimized cross-sectional shape. The Shape Optimization tool in ANSYS 14.0 is used to optimize the shape, which is then tested for design failure in ABAQUS 6.11. Shape optimization conserved 18.74 percent of the material while maintaining all of its strength and aerodynamic properties. This research opens up a numerical tool for improving future spoiler models in terms of mass reduction.

2.**LITERATURE SURVEY:**

[1] Artificial Neural Network (ANN) model was used to help cars dealers recognize the many characteristics of cars, including manufacturers, their location and classification of cars according to several categories including: Make, Model, Type, Origin, DriveTrain, MSRP, Invoice, EngineSize, Cylinders, Horsepower, MPG_Highway, Weight, Wheelbase, Length. ANN was used in prediction of the number of miles per gallon when the car is driven in the city(MPG_City). The results showed that ANN model was able to predict MPG_City with 97.50 % accuracy. The factor of DriveTrain has the most influence on MPG_City evaluation. Similar studies can be carried out for the evaluation of other characteristics of cars.

Artificial Neural Network for Forecasting Car Mileage per Gallon in the City

Afana, Mohsen; Ahmed, Jomana; Harb, Bayan; Abu-Nasser, Bassem S.; Abu-Naser, Samy S.

[2]The ability to model and predict fuel usage is critical for improving vehicle fuel economy and preventing fraud in fleet management. Internal factors such as distance, load, vehicle attributes, and driver conduct, as well as external factors such as road conditions, traffic, and weather, all influence a vehicle's fuel usage. However, not all of these variables may be measured or available for the analysis of fuel usage. We explore a scenario in which just a subset of the above elements is accessible as a multivariate time series from a long-distance public bus. As a result, the task is to model and/or anticipate fuel usage using only known data while capturing as much as possible indirect effects from other internal and external elements.Machine Learning (ML) is appropriate for this type of analysis since the model may be built by learning data patterns. We analyze the predictive abilities of three machine learning algorithms in estimating bus fuel consumption given all relevant characteristics as a time series in this study. In comparison to

both gradient boosting and neural networks, the random forest technique generates a more accurate forecast, according to the analysis.

Fuel consumption prediction of fleet vehicles using Machine Learning: A comparative study Sandareka Wickramanayake; H.M.N. Dilum Bandara.

3.THEORITICAL ANALYSIS:

Project Work Flow:

1. Data Collection
2. Data Pre-processing
3. Model Building
4. Application Building

3.1. HARDWARE AND SOFTWARE REQUIREMENTS IN THE PROJECT:

For running a machine learning model on the system you need a system with minimum of 16 GB RAM in it and you require a good processor for high performance of the model.
In the list of **software requirements** you must have:

- Jupyter Notebook for programming, which can be installed by Anaconda IDE.
- Python packages
- A better software for running the html and css files for application building phase e.g. spyder.

4.EXPERIMENTAL INVESTIGATIONS:

4.1 Data Preprocessing:-

(i) First check is there any null value in the dataset:-

```
In [5]: data.isnull().sum()

Out[5]: mpg             0
        cylinders       0
        displacement    0
        horsepower      0
        weight          0
        acceleration    0
        model year      0
        origin          0
        car name        0
        dtype: int64
```

(ii) Check whether the dataset contains numerical or categorical features:-

```
In [3]: data
```

Out[3]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52	2130	24.6	82	2	vw pickup
395	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
396	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
397	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

All are numerical features .There are no categorical features in the dataset. Therefore no need for Label Encoding and OneHot Encoding.

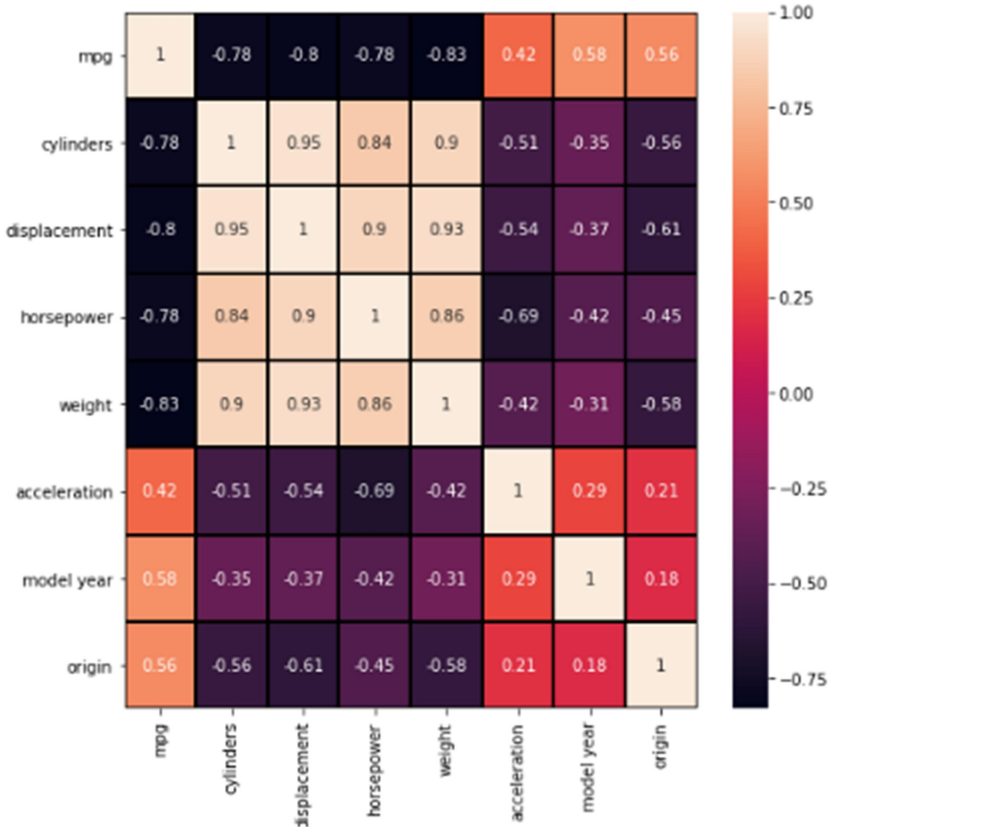
(iii) Dividing dataset into dependent and independent features:-

Shape of our data is 398 rows and 9 columns. The first column is the filename and the last column is the class label. x=data.iloc[:,1:8].
All the features except the file name and '**mpg**' are independent features.
y=data.iloc[:,0:1]
The '**mpg**' is the dependent feature.

Mileage per Gallon is mpg.

(iv) Plotting Correlation :-

```
In [10]: #importing necessary libraries.
         import matplotlib.pyplot as plt
         import seaborn as sns
         sns.heatmap(data.corr(),annot=True,linecolor ='black', linewidths = 1)
         fig=plt.gcf()
         fig.set_size_inches(8,8)
```



(v) Training and testing data:-

We have splitted our data such that 90% of our data will be used for training cases and 10% for testing cases.

4.2 Score Parameters:-

For checking how good our model is, we have used three scoring parameters.

1. Mean Squared Errors
2. Mean Absolute Errors
3. r2_score.

Mean Squared Error: In statistics, the mean squared error or mean squared deviation of an estimator measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value.

Mean Absolute Error: In statistics, mean absolute error is a measure of errors between paired observations expressing the same phenomenon. Examples of Y versus X include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative technique of measurement.

R2_Score: In statistics, the coefficient of determination, denoted R^2 or r^2 and pronounced "R squared", is the proportion of the variation in the dependent variable that is predictable from the independent variable.

```
In [34]: def scores(y_test,ypreds):
             mse=mean_squared_error(y_test,ypreds)
             print('Mean Squared: ',mse)
             mae=mean_absolute_error(y_test,ypreds)
             print('Mean Absolute Error: ',mae)
             accuracy=r2_score(y_test,ypreds)
             print('R2_score: ',accuracy )
```

4.3 Used Algorithms and Methods:-

1. **Multilinear Regression:-**

   Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome

of a response variable. Multiple regression is an extension of linear (OLS) regression that uses just one explanatory variable.

```
In [35]: lr=LinearRegression()
         lr.fit(x,y)

Out[35]: LinearRegression()

In [36]: ypredlr=lr.predict(x_test)

In [37]: print(ypredlr)

         [13.00967075 24.17837774 11.40931118 20.84102122 17.40733067 29.79045882
          33.52610844 23.51876689 14.6688346  27.08881834 33.87922031 34.08456234
          21.30815307 26.04501146 16.2341128  30.96404189 28.68409287 28.97095081
          17.3760003  31.05112323 15.76987547 24.67257256 27.02340786 20.35100415
          29.75181158 28.62077861 31.20089836 30.48359625 29.90190444 17.94052359
          20.43457266 31.43628126 20.72090784 32.24921759 23.92860085 26.01285408
          21.20921058 16.8478706  32.2777473   9.0000764 ]

In [38]: scores(y_test,ypredlr)

         Mean Squared:  9.697403780857837
         Mean Absolute Error:  2.4946514519331386
         R2_score:  0.859700644271482

In [39]: lr.score(x_test,y_test)

Out[39]: 0.859700644271482
```

```
In [39]: lr.score(x_test,y_test)

Out[39]: 0.859700644271482

In [40]: lr.intercept_

Out[40]: -17.284999204176728

In [41]: lr.coef_

Out[41]: array([-0.43241036,  0.01969494, -0.01631524, -0.00661211,  0.08119764,
                 0.75194963,  1.44272293])
```

2. Polynomial Regression:-

In statistics, polynomial regression is a form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modeled as an nth degree polynomial in x.

```
In [42]: from sklearn.preprocessing import PolynomialFeatures

In [43]: poly = PolynomialFeatures(interaction_only=True)
         x1=x
         poly.fit_transform(x1)

Out[43]: array([[1.0000e+00, 8.0000e+00, 3.0700e+02, ..., 8.4000e+02, 1.2000e+01,
                 7.0000e+01],
                [1.0000e+00, 8.0000e+00, 3.5000e+02, ..., 8.0500e+02, 1.1500e+01,
                 7.0000e+01],
                [1.0000e+00, 8.0000e+00, 3.1800e+02, ..., 7.7000e+02, 1.1000e+01,
                 7.0000e+01],
                ...,
                [1.0000e+00, 4.0000e+00, 1.3500e+02, ..., 9.5120e+02, 1.1600e+01,
                 8.2000e+01],
                [1.0000e+00, 4.0000e+00, 1.2000e+02, ..., 1.5252e+03, 1.8600e+01,
                 8.2000e+01],
                [1.0000e+00, 4.0000e+00, 1.1900e+02, ..., 1.5908e+03, 1.9400e+01,
                 8.2000e+01]])

In [46]: poly_reg_model.fit(x1, y)

Out[46]: LinearRegression()

In [47]: ypredpoly = poly_reg_model.predict(x_test)

In [48]: ypredpoly

Out[48]: array([13.00967075, 24.17837774, 11.40931118, 20.84102122, 17.40733067,
                29.79045882, 33.52610844, 23.51876689, 14.6688346 , 27.08881834,
                33.87922031, 34.08456234, 21.30815307, 26.04501146, 16.2341128 ,
                30.96404189, 28.68409287, 28.97095081, 17.3760003 , 31.05112323,
                15.76987547, 24.67257256, 27.02340786, 20.35100415, 29.75181158,
                28.62077861, 31.20089836, 30.48359625, 29.90190444, 17.94052359,
                20.43457266, 31.43628126, 20.72090784, 32.24921759, 23.92860085,
                26.01285408, 21.20921058, 16.8478706 , 32.2777473 ,  9.0000764 ])

In [49]: scores(y_test,ypredpoly)

         Mean Squared:  9.697403780857837
         Mean Absolute Error:  2.4946514519331386
         R2_score:  0.859700644271482
```

3. Decision Trees Regression:-

Decision trees build regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

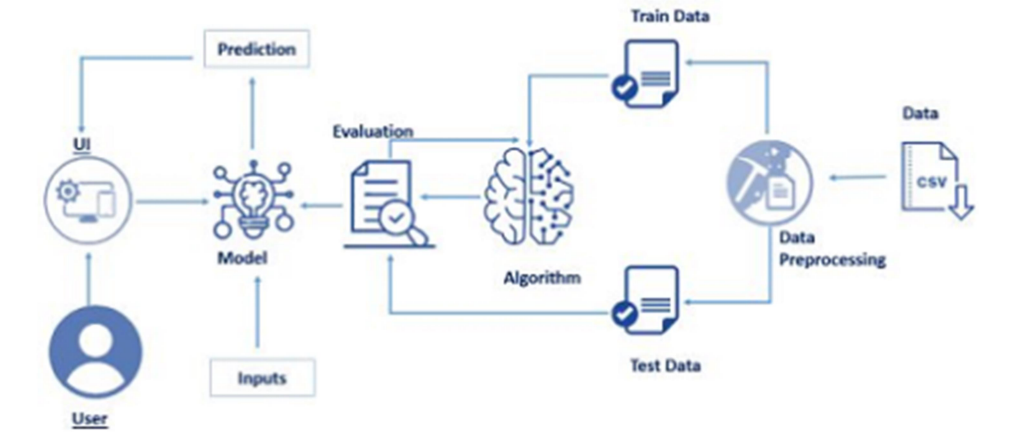
```
In [50]: from sklearn.tree import DecisionTreeRegressor

In [51]: dtr = DecisionTreeRegressor()
         dtr.fit(x,y)

Out[51]: DecisionTreeRegressor()

In [52]: ypreddt = dtr.predict(x_test)

In [53]: print(ypreddt)

         [14.  25.  13.  21.  18.  35.  34.1 20.  15.  23.5 40.9 37.2 18.  23.
          15.5 35.7 31.  27.  18.  37.3 15.5 23.  24.  18.  34.5 25.4 36.1 34.
          30.  16.  18.6 37.  15.  33.5 22.4 24.  19.  16.9 31.9 12. ]

In [ ]:

In [54]: scores(y_test,ypreddt)

         Mean Squared:  0.0
         Mean Absolute Error:  0.0
         R2_score:  1.0
```

4. Random Forest Regression:-

Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.

```
In [55]: from sklearn.ensemble import RandomForestRegressor
         rf = RandomForestRegressor(n_estimators=10,random_state=0)
         rf.fit(x,y)
         ypredrf=rf.predict(x_test)
         scores(y_test,ypredrf)

         Mean Squared:  1.0828725000000003
         Mean Absolute Error:  0.6717500000000003
         R2_score:  0.9843333001781338

In [56]: ypredrf=rf.predict(x_test)

In [57]: ypredrf

Out[57]: array([14.  , 24.8 , 14.75, 20.92, 17.9 , 32.9 , 35.16, 20.5 , 14.9 ,
                23.83, 41.09, 38.5 , 18.02, 23.6 , 15.45, 34.62, 30.39, 26.59,
                17.85, 36.22, 15.55, 23.35, 24.  , 18.1 , 35.2 , 25.75, 36.03,
                32.95, 30.72, 15.8 , 18.78, 34.72, 16.82, 34.8 , 21.25, 24.3 ,
                18.84, 16.5 , 35.78, 12.1 ])

In [58]: scores(y_test,ypredrf)

         Mean Squared:  1.0828725000000003
         Mean Absolute Error:  0.6717500000000003
         R2_score:  0.9843333001781338
```

5. FLOW CHART:-



6. RESULTS:-

The score parameters of different algorithms are displayed below:

1. Multiple Linear Regression:-

```
In [38]: scores(y_test,ypredlr)

         Mean Squared:  9.697403780857837
         Mean Absolute Error:  2.4946514519331386
         R2_score:  0.859700644271482
```

2. Polynomial Regression:-

```
In [49]: scores(y_test,ypredpoly)

         Mean Squared:  9.697403780857837
         Mean Absolute Error:  2.4946514519331386
         R2_score:  0.859700644271482
```

3. Decision Tree Regression:-

```
In [54]: scores(y_test,ypreddt)

         Mean Squared:  0.0
         Mean Absolute Error:  0.0
         R2_score:  1.0
```

4. Random Forest Regression:-

```
In [62]: rf = RandomForestRegressor(n_estimators=300,random_state=42)
         rf.fit(x,y)
         ypredrf=rf.predict(x_test)
         scores(y_test,ypredrf)

         Mean Squared:  0.5471064500000002
         Mean Absolute Error:  0.5147166666666705
         R2_score:  0.9920846152037688
```

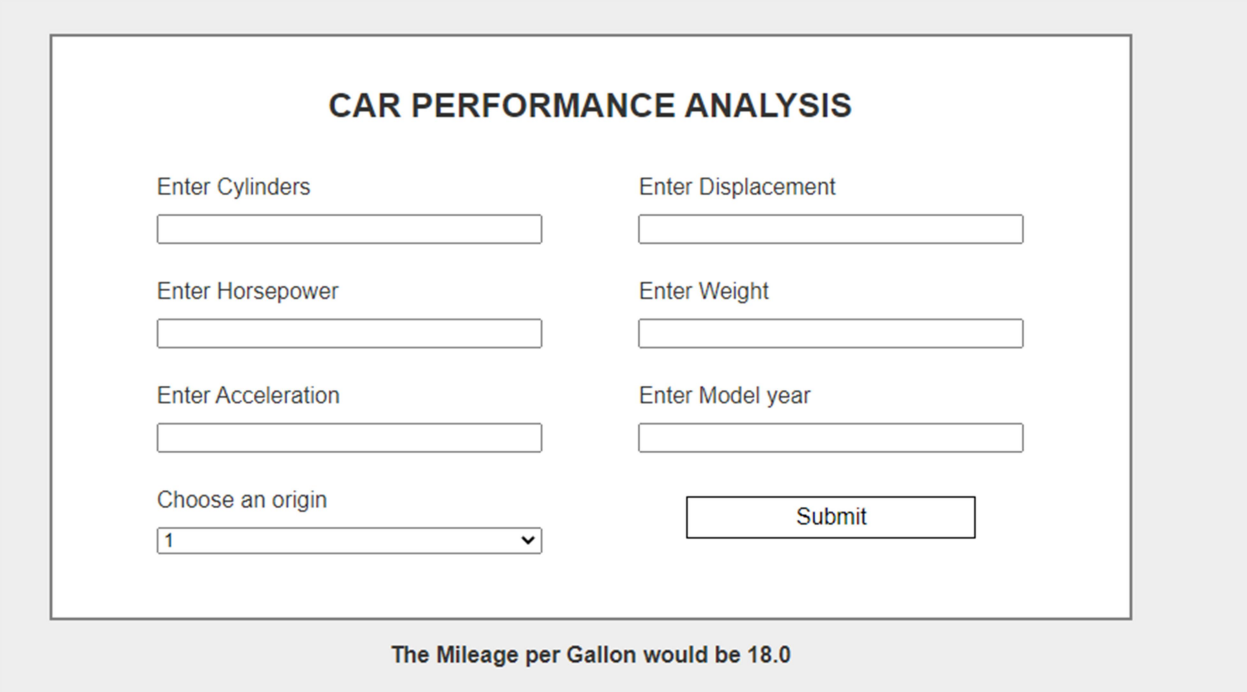
Therefore, decision tree regression gives the best results. It has the least errors and the maximum scores among the four methods used.

Thus, for further Flask App and IBM Watson Cloud Deployment, a pickle model of decision tree algorithm will be created.

```
import pickle
pickle.dump(dtr,open("Car.pkl","wb"))
```

7. APPLICATIONS:-

The prediction system was deployed on IBM Watson Cloud and a web application was created for the prediction system. In the application we have to enter the input values such as the number of cylinders the vehicle contains, its displacement , its horsepower and its acceleration. After giving these inputs, Mileage per Gallon (mpg) will be displayed based on the calculations.



With the above parameters we can predict the performance of the vehicle. And with the given ranges of mileage and horsepower we can estimate where and in what kind of activities are best suited for the vehicle.

For example a low power high mileage could be used as a taxi fare system whereas a high power low mileage system could be used for heavy lifting such as JCBs and tractors.
Also the prediction system could be used to determine the health of a car and to estimate its resale value. To find out how much of a depreciation has happened to vehicle and what not

8.CONCLUSION-

Therefore, after predicting performance of the car prediction system, it was deployed on IBM Watson Cloud and a web application was launched using Flask.

A Video Demonstration Link has been added to enhance our project:-

https://www.youtube.com/watch?v=Wc9LXpJJ_QI

9. REFERENCES:-

[1] Artificial Neural Network for Forecasting Car Mileage per Gallon in the City
Afana, Mohsen; Ahmed, Jomana; Harb, Bayan; Abu-Nasser, Bassem S.; Abu-Naser, Samy S.

[2] Fuel consumption prediction of fleet vehicles using Machine Learning: A comparative study by Sandareka Wickramanayake; H.M.N. Dilum Bandara.